

REMARKS

Claims 1, 3 to 14, and 16 to 28 are pending in this application. Of these, claims 1 and 14 are independent.¹ Favorable reconsideration and further examination are respectfully requested.

All of the claims were rejected over U.S. Patent Publication No. 2002/0168621 (Cook) in view of U.S. Patent Publication No. 2006/0059253 (Goodman). This rejection was traversed in a reply filed on May 9, 2007. An advisory action issued on May 25, 2007, which maintained the rejection in view of Cook and Goodman. In response, Applicants are submitting this reply.

Independent claim 1 has been amended to recite a method, performed by one or more processing devices, for use in an electronic learning system that stores information as learning objects. The method comprises designating a target learning object as a project object, and storing version dependency data in the project object. The version dependency data identifies at least a version of a first object upon which the project object directly depends, and a version of a second object upon which the project object indirectly depends. The first object stores dependency data identifying the second object upon which the first object depends. The first object does not store version dependency data identifying the version of the second object upon which the first object depends.

As explained in the prior amendment, Cook fails to disclose or to suggest the features of claim 1, in particular, storing version dependency data in the project object, where the version dependency data identifies at least a version of a first object upon which the project object directly depends, and a version of a second object upon which the project object indirectly

¹ The Examiner is urged to independently confirm this recitation of the pending claims.

depends. In fact, it was admitted, on page 3 of the Office Action, that Cook does not disclose version dependency data. Therefore, Cook could not possibly disclose or suggest the foregoing features. Nevertheless, the Advisory Action cites paragraphs 77 to 81 of Cook for allegedly disclosing "version dependency". In this regard, the cited paragraphs of Cook do not even disclose dependencies between objects (in fact, the word "depend" is not even used in these paragraphs). Applicants are therefore at a loss as to how those portions of Cook could possibly disclose or suggest the foregoing features of claim 1.

Applicants do note the description of the student object in paragraph 83, and subsequent paragraphs, such as 0091, where the student object is referenced by an agent. Applicant further note that Fig. 2 shows a client system, which may be used by a student defined by an object. According to Cook,

In cases of low-bandwidth network access, the client system can have one or more disc drives 209 which can be used as a pre-fetch buffer or a read-only cache. The disks preferably are magnetic, although in a less preferable embodiment they can also include CDRoms. This optional capability serves to enhance communications efficiency in cases where the network has relatively low bandwidth. Large files can be downloaded in advance of a student session or the student client can cache read-only data across sessions obviating the need for downloading such files. Such caching requires the operating system components *to maintain some form of version control of the read-only data.* (0097) (emphasis added)

Thus, while Cook does describe moving data objects between locations (portable media and server), and does mention version control briefly and generally, there is nothing in Cook that discloses or suggests storing version dependency data in one type of object -- a project object -- and dependency data, but not version dependency data, in other types of objects.

As previously described, Goodman describes a system that keeps track of different versions of software, e.g., through version control services, as described below.

As with standard development code, when media content is created and edited, the version control services maintain a revision history of changes. This way, if it is necessary to revert to an original piece of media content, it is not necessary to go all the way back to the original source (which in the case of finding an image in a CD-ROM library containing 10,000 images, for example, could be a difficult task). In practice, this may mean storing the original and final copies of media (especially where volume is an issue). For this reason, a process for managing multiple versions of media content is put into place in the preferred development architecture 500.²

Paragraph 0234 describes version control as follows:

Version control and compatibility are key considerations when managing these packages. Note that version control not only applies to software components, but also to all components of a given package, including test scripts, test data and design documentation. It is also of great importance to keep track of which version is in which environment. If incompatibilities are discovered, it must always be possible to "roll back" to a previous consistent state; that is, to revert to an earlier version of one or more components. It must be possible to define releases of a configuration—a list of version numbers, one for each component of the package, which, together, form a consistent configuration. The smallest unit that can be version-controlled should be the package as defined in the packaging plan. This ensures that the lowest common denominator in all version-control activities is managed at the package level.

The Office Action and Advisory Action cite paragraphs 316 and 378 of Goodman for allegedly describing storage of version dependency data in a project object and dependency data, but not version dependency data, in objects that depend from the project object. These paragraphs are reproduced below for discussion purposes.

[0316] As illustrated in FIG. 26, the information management tools 602 is also responsible for object management 646. Object management tools provide capabilities for viewing objects, their methods and attributes, and the dependencies between these objects. Object management tools also provide specific analysis tools, in order to understand interdependencies between the core classes and the components. When classes and components are modified, impact analysis tools are required to see where the modified entity is being used, allowing them to understand what is the overall impact of the change. This is more complex than with traditional systems as a veritable spider's web of dependencies between classes, components, and applications may ensue. In addition, OO features such as inheritance and polymorphism make tracking down dependencies with simple text search tools much more difficult.

² Paragraph 0176

[0378] Process modeling tools 600 provide a graphical depiction of the business functions and processes being supported by a system. The tool(s) selected must support the modeling techniques being used in the development methodology; these include process decomposition, data flow and process dependency. Process modeling is a method for clarifying and communicating the business design of the system. The process model can provide an effective means of bringing people together, creating a shared vision of how the business is to function. A process model provides a means of standardizing a set of similar processes, which exist, for example, at different branches of the business.

As explained before, paragraph 0316 does describe dependencies among objects, and "specific analysis tools...to understand interdependencies between the core classes and the components".

Paragraph 0378, on the other hand, merely mentions "process dependency". Significantly, however, neither these paragraphs, nor the remainder of Goodman, describes exactly how the dependencies are analyzed or managed in connection with different versions. Furthermore, there is no disclosure or suggestion in paragraphs 0316 or 0378 (or elsewhere in Goodman) of storing version dependency data in one type of object – a project object – and dependency data, but not version dependency data, in other types of objects.

For at least the foregoing reasons, Applicants submit that even if Goodman were combined with Cook in the manner suggested, the resulting hypothetical combination would fail to disclose or to suggest the features of claim 1. Independent claim 14 is a program claim that roughly corresponds to claim 1, and is also believed to be patentable.

If the Examiner persists in maintaining the above rejection, the Examiner is respectfully requested to show *precisely* where the features are disclosed – i.e., the exact text. Merely reciting the claim language and then identifying a paragraph number in the references is not sufficient. Applicants also respectfully request for the Examiner to provide appropriate reasoning so that, if an appeal to the board becomes necessary, Applicants will have a full

understanding of the Examiner's case, and be able to identify what specifically in the text the Examiner is relying upon, and how that text is being construed.

The remaining dependent claims are also believed to define patentable features of the invention. Each dependent claim partakes of the novelty of its corresponding independent claim and, as such, each has not been discussed specifically herein.

It is believed that all of the pending claims have been addressed. However, the absence of a reply to a specific rejection, issue or comment does not signify agreement with or concession of that rejection, issue or comment. In addition, because the arguments made above may not be exhaustive, there may be reasons for patentability of any or all pending claims (or other claims) that have not been expressed. Finally, nothing in this paper should be construed as an intent to concede any issue with regard to any claim, except as specifically stated in this paper, and the amendment of any claim does not necessarily signify concession of unpatentability of the claim prior to its amendment.

In view of the foregoing amendments and remarks, Applicants respectfully submit that the application is in condition for allowance, and such action is respectfully requested at the Examiner's earliest convenience.

Applicants' undersigned attorney can be reached at the address shown below. All telephone calls should be directed to the undersigned at 617-521-7896.

Applicants: Wolfgang Theilmann, et al.
Serial No. : 10/809,873
Filed : March 25, 2004
Page : 15 of 15

Attorney's Docket No.: 13909-161001
Client Ref: 2004P00116US

Please apply any fees or credits due in this case to Deposit Account 06-1050 referencing
Attorney Docket No. 13909-161001.

Respectfully submitted,

Date: _____

June 26, 2007



Paul A. Pysher
Reg. No. 40,780

Fish & Richardson P.C.
225 Franklin Street
Boston, MA 02110-2804
Telephone: (617) 542-5070
Facsimile: (617) 542-8906

161.doc